A Systematic Approach to Developing Safe Tele–operated Robots *

Diego Alonso, Pedro Sánchez, Bárbara Álvarez, Juan A. Pastor

Universidad Politécnica de Cartagena, Division of Systems and Electronic Engineering (DSIE) Campus Muralla del Mar, s/n. Cartagena, E-30202, Spain
e-mail: diego.alonso@upct.es

Abstract. Tele–operated service robots are used for extending human capabilities in hazardous and/or inaccessible environments. Their use is undergoing an exponential increase in our society, reason why it is of vital importance that their design, installation and operation follow the strictest possible process, so that the risk of accident could be minimised. However, there is no such process or methodology that guides the full process from identification, evaluation, proposal of solutions and reuse of safety requirements, although a hard work is being done, specially by the standardisation committees. It's also very difficult to even find in the literature examples of safety requirements identification and use. This paper presents the engineering process we have followed to obtain the safety requirements in one of the robots of the EFTCoR[1] project and the way this requirements have affected the architecture of the system, with a practical example: a crane robot for ship hull blasting.

1 Introduction

Human operators use tele–operated service robots for performing more or less hazardous operations (manipulation of heavy and/or dangerous products) in more or less hostile environments (nuclear reactors, space missions, warehouses, etc). Anyway, independently of the operation, the robot has to interact with both the environment it's working on and with human operators. So, it is essential that the design (which include both software and hardware) of the robot involves no (or an acceptable level of) risk, neither for the operators, nor for the environment nor for the robot itself.

Nevertheless, it's not always possible to make a system free of failures in its design or operation. Apart from the risk inherent to the use of the mechanisms themselves, these systems work in hazardous environments, where the

* This work has been partially supported by the Spanish Government programs CICYT, ANCLA (TIC2003-07804-C05-02), part of DYNAMICA (DYNamic and Aspect-Oriented Modeling for Integrated Component-based Architectures)

[1] Project EFTCoR: Environmentally Friendly and Cost-Effective Technology for Coating Removal. Fifth framework programme of the European Community for research, key action Competitive and Sustainable Growth (GRD2-2001-50004)

probability of the risk is higher than normal. Should a failure happen, the consequences of it can even involve the loss of human lives. [1] documents many cases of computer–related failures, such as the Therac–25 (a radiation–therapy device), the missiles shield in Saudi Arabia, etc.

But safety aspects are seldom included in the design process of the system from the beginning, even though they are a critic aspect. Generally, safety has to conform and adapt to the already designed system and not vice versa, when it's known that safety involves not only the design of the software but also the hardware. In fact, there are many situations in which a simple hardware solution can eliminate a hazard or simplify the design of the safety software.

However, the identification of safety requirements is not different from the identification of the rest of requirements of the system. It only requires a more thorough study, due to their importance (don't forget, human lives and equipment integrity may depend on it!). On the other hand, safety has a big repercussion in the design phase, specially when the time to define the architecture of the system arrives. Its impact is even bigger by the need to avoid *common failure modes*, that can propagate failures within different units of the system.

The objectives of this paper are to stress the importance of the capture of the safety requirements early in the design process and to present a practical experience on how to capture these safety requirements and how they can alter the design of the system. The example presents a thorough study of the safety requirements that a crane robot (a member of the EFTCoR [2,3] project) must conform to in order to work in such a hazardous environment as shipyards are. The EFTCoR project is about to end after three years of intense work. Altough the robot fulfils the basic safety requirements, we are now thinking about making a commercial version of it, so a deeper study of safety is needed.

This paper is structured in five sections. Section 2 presents a brief description of the EFTCoR project and the safety characteristics that make it a perfect example. In section 3 the process followed to obtain the safety requirements is commented, while section 4 presents the process of identification of safety requirements for the EFTCoR crane robot. Finally, section 5 summarises the contents of the paper and outlines future lines of work.

2 EFTCoR: the Danger of Cleaning Ship Hulls in Shipyards

The EFTCoR family of robots offers a global solution to the problems related to the most dangerous hull maintenance operations, such as cleaning, blasting and painting (see Fig. 1-a). The solution is provided by means of two families of robots: tele–operated cranes and climbing vehicles, depending on the working area. All these robots consist of a primary positioning system, capable of covering large hull areas, and a secondary positioning system, mounted on the primary system, that can position a tool over a relatively small area (4 to 16 m^2). The robots have been developed to achieve the objective of performing the current hull cleaning operations in a way that avoids the emissions of residues to

the environment and enhances the working conditions of the shipyard operators without worsening the current costs and operation times. Figure 1-b shows the crane robot in action.

a) Before EFTCoR b) With EFTCoR crane

Fig. 1. Blasting operation

The design of such a complex system as EFTCoR involves the necessity of early detection and identification of failures so that correcting measures can be adopted early in the design of the robot. The fundamental characteristics of the EFTCoR that makes it necessary to take into account the need of a safe approach when designing the robots are summarised by the following points:

- ▷ The operator uses a heavy mechatronic device whose range of movement can cause serious damage (see Fig. 1-b).
- ▷ The system has to be used outdoors, so it has to be able to deal with atmospheric agents that can alter its normal operation (rain, water on the ground, dust, noise, wind, etc).
- ▷ The working environment of the robots (shipyards) is very dynamic: there are many cranes, load and unload of heavy equipments, lots of operators moving around (either working on the robot or conscious or not of its presence), etc.
- ▷ Some maintenance operations include the blasting of the hull with high–pressure abrasive particles. The energy of the jet makes it very dangerous for human operators and for the rest of the equipment, so it's absolutely necessary to train operators in the use of the tool, to maintain the equipment in perfect conditions and to install all the security components needed. Also, as a result of the impact of the jet with the hull, a lot of dust is produced, worsening the condition of the working place.

3 A Safety Process

The purpose of this section is to present a brief summary of the steps we have followed for discovering the safety requirements for the EFTCoR and the consequences they imply on the architecture of the system. To work this out we

have based our work on the ANSI standard for robotics [4] (see next point), completing it with the contribution of other authors, such as Douglass [5], that complete the proposal.

Before going on, we introduce the meaning of some words that are used in the paper. According to Douglass, a *risk* is an event or condition that can occur but is undesirable; *safety* is the characteristic of a system that does not incur too much risk to persons or equipment and an *accident* is damage to property o harm to persons, the happening of a risk. A *safety system* is, according to the definition of ANSI/RIA, a system that has been tested, evaluated and proven to operate in a reliable and acceptable manner when applied in a function critical to health and welfare of personnel. Leveson [6] defines a *hazard* as a state or set of conditions of a system (or object) that, together with other conditions in the environment of the system (or object) will inevitably lead to an accident (loss event).

3.1 Survey of Safety Standards and Techniques

There are several approaches to manage safety in literature. Many deal with the problem of designing a standard that guides the whole process (from identification to solution) while others are simple tools or techniques. Among the standards we want to stress the European Standard EN 61508:2001 [7] and the American ANSI/RIA R15.06-1999 [4]. Among the techniques for safety designs we highlight fault trees [8] and ROPES [5] (*Rapid Object-oriented Process for Embedded Systems*).

EN 61508:2001. This European standard sets up a generic approximation for dealing with all the activities related to the life–cycle of the systems that use electric and/or electronic and/or programmable devices for safety functions. The other main purpose of this standard is to serve as basis for the development of specific standards for each application sector, that would take into account techniques and solutions typical of the sector.

ANSI/RIA R15.06-1999. The objective of this standard is to enhance the safety of personnel using industrial robot systems by establishing requirements for the manufacture (including remanufacture and overhaul), installation, safeguarding methods, maintenance and repair of manipulating industrial robots. It is the intention of this standard that the manufacturer (including remanufacturer and rebuilder), the installer and the end–user have specific responsabilities.

Fault trees. It's one of the most popular approaches to identify, evaluate and manage safety requirements. These trees provide a graphical notation and a formal support that makes it easy to make the analysis from the perspective of the system failures and their origins. However, they do not offer a global framework for requirement specification as a discipline.

ROPES. ROPES is, in words of Douglass, "a development process that emphasises rapid turnaround, early proofs of correctness and low risk". It's an iterative process that makes the design in small, incremental steps. Douglass

proposes an eight–steps methodology for dealing with the safety aspects of any system.

3.2 Process of Elicitation of Requirements

As last section shown, until a new standard derived from EN 61508 and targeted to robotics appear, only the ANSI standard offers an specific guide to this kind of systems. But ANSI encourages the use of hardware solutions (such as barriers, light beams, buttons, etc), and does not provide any guide on the use of more complex, software based, solutions.

To complete this lack of detail, the proposal "eight steps to safety" from Douglass has been adopted. In it, Douglass proposes some design patterns oriented to the achievement of a particular safety objective, such as *multi–channel voting pattern, watchdog pattern, safety executive pattern,* etc. By using these patterns we can design software solutions that conform to the needs imposed by the ANSI standard, according to the level of risk of a particular hazard.

Finally, the technique of *fault trees* can be used to obtain the possible causes of the failures that are analysed in the second step of the methodology we propose. Fault trees is a very used and mature technique, but it doesn't help in neither measuring nor classifying nor solving failures. We haven't use this technique for obtaining the causes of the failures, although we think it would have been a good idea to do so.

The four–steps methodology we present proposes the fusion of the standards and techniques presented in subsection 3.1. It encourages the tracking of safety throughout the life–cycle of the robot (as EN 61508 proposes) and uses the ANSI standard as a guide to classify hazards and to propose solutions. By completing ANSI with the contributions of Douglass, it is possible to deal with the design of software–based solution that are more complex than a simple barrier.

Step 1 ► Identify hazards. It is desirable that a system should normally work without imminent hazards. So, the first step is to identify all the tasks that involve the use of the system and that have potential hazards. After that, for each task an analysis of the hazards is performed. Some possible sources for the identification of hazards, that can serve as a starting point in their identification, are the following ones (extracted from [4]):

- The movement of mechanical components, especially those which can cause trapping or crushing.
- Stored energy in moving parts, electrical or fluid components.
- Power sources: electrical, hydraulic, pneumatic.
- Hazardous atmospheres, material or conditions: explosive or combustible, radioactive, high temperature and/or pressure, etc.
- Acoustic noise, vibrations, EMI, etc.
- Human failures in design, construction, installation, and operation, whether deliberate or not.

This analysis of hazards also include the identification of the possible causes of the failure (hardware, software or human), the task in which it can happen, the reaction to the happening of the hazard, and some temporal data (adopted from Douglass) relative to how long can the hazard be tolerated before it results in an accident (tolerance time), the maximum amount of time to detect the happening (detection time) and the maximum time to react to it (reaction time).

Step 2 ► Identify risks. The objective of this second step is to identify the possible risks of the system and classify them, according to the impact they have on the environment and linking them to the hazards identified on the the first step. The ANSI standard says that, for each risk, three characteristics have to be evaluated. They are the level of severity, the level of exposure and the level of avoidance, each with two different values (for a total of eight possible combinations). Depending on the different values of these characteristics, a Risk Reduction Category (RRC) is obtained. Based on the RRC, ANSI requires a certain level of performance of the safeguard and circuit that are to be design to reduce the risk (simple, single channel, single channel with monitoring and control reliable). Moreover, ANSI also recommend the adoption of safety policies to help human operators avoid some risks (training, presence detectors, security barriers, etc).

After applying the safeguards designed for the specific RRC of the risk, a new analysis is performed to calculate the residual risk, just to be sure that the risk is kept at a tolerable level for both the system and the environment. This process does not end here but has to be repeated during the life–cycle of the robot to ensure that no new risk appears and that the risk already identified are kept under control.

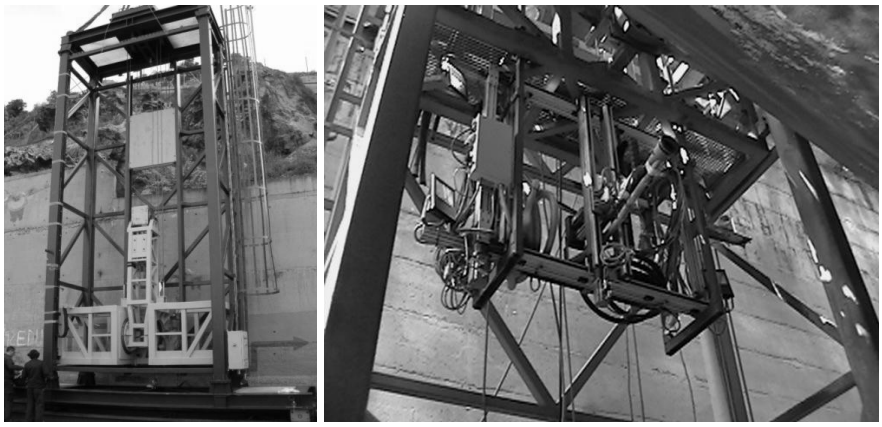
Step 3 ► Specify safety requirements. The purpose of this third step is to extract the safety requirements for the system from the results of the previous steps. This is quite difficult to do, because neither ANSI nor Douglass offer a methodology to deduce the requirements from the previous results, so this extraction has been handmade. At this point, it's necessary to have an appropriate process for the harvest of requirements, a way to catalogue them so that they can be reused in other systems of the domain of application (tele–operated robots in our case), as well as tools for tracking the use of the requirements throughout the development process and, in particular, until the architecture of the system. This is the kind of work the Universidad de Murcia is doing inside DYNAMICA.

Step 4 ► Make safe designs. The design of the architecture of the system must consider the safety measures and avoid that the failure in a part spread through the rest of the system. A safe design must start off with the previous requirements of security (third step) to adopt a concrete architectural pattern that could be periodically reviewed when new hazards are identified. To be able to do it, to be able to be adaptable, a rigorous architectural approach that allows

the evolution of the architectural model due to new requirements or by evolution of the conditions of work is necessary.

4 Safety in the EFTCoR Project

In this section we present an example of the application of the process to obtain the safety requirements for the crane robot of the EFTCoR project. The crane robot uses a commercial crane as the primary positioning system (see Fig. 2-a) and a XYZ table as the secondary positioning system (see Fig. 2-b). The crane has its own control (provided by the manufacturer), a height of twelve meters and a weight of twenty tons, which make unavoidable the movement of the robot with the consideration of safety requirements. It also has, in its central zone, an articulated arm of two tons for holding the XYZ table (which includes a cleaning tool). The control system of the XYZ table has been designed to follow the cleaning instructions from a human operator or from a computer vision system, which finds the areas of the hull to be blasted.



a) Crane b) XYZ table

Fig. 2. Crane robot for cleaning vertical surfaces in EFTCoR

Due to the extension of the work, only the results of the safety analysis for the primary position system (tasks, hazards and risks) will be presented (see subsection 4.1). Subsection 4.2 presents the solution adopted for the hazard *"The arm of the primary system does not stop"* (see table 2, H13).

4.1 Identification of Hazards and Risks for the Primary System

Using the functional requirements of the EFTCoR system as a starting point, a total of 30 different tasks with a potential hazard have been identified (excerpt

in Table 1). These tasks are performed not only by the operator of the robot but also by the maintenance and cleaning staff, can have been planned or not, and their frequency can be daily, weekly, monthly, annually, etc. Table 2 shows an excerpt of the 31 hazards related to the tasks to be performed by the robot (only the hazards related to the primary are shown). These two tables comprise the first step.

#	Type	Description
T1	Operator	Move the primary (rail)
T2	Operator	Move the primary (vertical axis)
T8	Operator	Execute a sequence
T20	Maintenance	Calibrate one of the axes of the primary
T23	Maintenance	Repair an axis (primary or secondary)

Table 1. Excerpt of tasks related to the primary system

Hazard	Risk	Origin	Prob.	Reaction
H3.Person in the rail	Very severe	There's a person standing on the rail	Med.	Raise alarm. Stop the primary. Emergency stop
H4.Obstacle in the rail	Severe	There's an obstacle on the rail	Med.	Raise alarm. Stop the primary. Emergency stop
H5.Obstacle in the vertical axis	Very severe	There's an obstacle on the trajectory	High	Raise alarm. Stop the primary. Emergency stop
H7.The limit switch of the vertical axis is passed	Very severe	Sensor or software error. Comm failure	Low	Raise alarm. Emergency stop
H8.The limit switch of the rail is passed	Very sever	Sensor or software error. Comm failure	Low	Raise alarm. Stop power source
H13.The arm of the primary system does not stop	Very severe	Joint control error. Comm or power failure	Low	Raise alarm. Stop power source. Emergency stop
H15.The sequence of the primary does not end	Very severe	Sequence control error. Comm failure	Low	Raise alarm. Stop primary

Table 2. Excerpt of hazards related to the primary system

Finally, Table 3 shows the results of the step identify risks, but just for the hazards related to the primary positioning system (with the consequences of an

accident, the RRC required according to ANSI, the safeguard adopted and the residual RRC).

#	Risk	RRC	Solution	RRC
H3	Run over a person	R2A	Add presence sensors to the rail. Add an acoustic signal when the robot moves	R3B
H4	Damage obstacle and primary	R2A	Add presence sensors to the rail. Add an acoustic signal when the robot moves	R3B
H5	Damage obstacle and primary	R1	Add presence sensors to the vertical axis.	R3B
H7	Damage to equipment or primary or persons	R2B	Add mechanic limits	R4
H8	Damage to equipment or primary or persons	R2B	Add mechanic limits	R4
H13	Damage to equipment or primary or persons	R2B	Add an emergency stop mechanism. Add sensors external to the control loop	R4
H15	The robot can even knock over	R2B	Add an emergency stop mechanism. Add sensors external to the control loop	R4

Table 3. Excerpt of solutions for the primary system hazards

4.2 Analysis of a Hazard: "The Arm of the Primary System does not Stop"

An analysis with detail of this hazard takes us to associate the following possible sources of error: (1) any sensor integrated with the motors that move the arm fails; (2) the electric power is off and (3) the control unit does not run correctly (a hardware fail or a software error). The hazard H13 may imply the breaking of mechanical parts, the precipitation of components to the floor or damages to the human operator. See table 2 and table 3 for the characterisation of this hazard.

Following the ANSI standard, the levels of the severity of the injury, the frequency of the exposure and the probability of avoidance are evaluated. This evaluations results in a RRC of R2B. Figure 3 shows the deployment partitioning of the system (using an extension of the standard UML notation) that accomplishes the R2B to R4 risk reduction for the hazard H13. This particular solution uses the *watchdog pattern* from Douglass [5]. The limitation of space in this paper does not allow us to give all the details related to the real implementation of the safeguard for this hazard, although table 4 shows the connection between the entities shown in the deployment diagram and their implementation in Ada. Anyway, the full description of the solution follows:

1. When a movement command is received, the `Man Machine Interface` (MMI) node forwards it simultaneously to the `Control Unit` node (that will execute it) and to the redundant node, which is in charge of detecting possible hazards (`Safety Control` node).

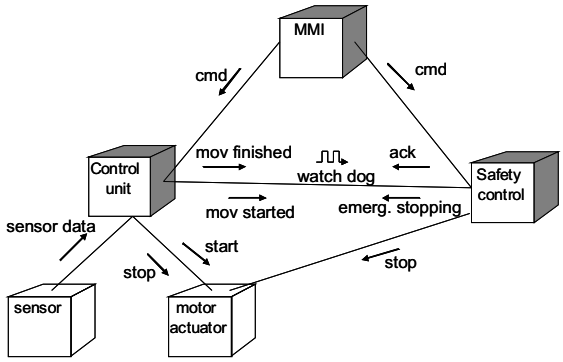


Fig. 3. Deployment diagram for H13

2. The `Control` node reads periodically the current position of the joint from a sensor and controls the actuator. The `Safety` node is in charge of stopping the motor when it detects that the motor is not working properly.
3. Just before the execution of any command, the `Control` node sends a message to the `Safety` node, authorising the start of the movement. From here, the `Control Unit` sends to the `Safety` node the current value just read from the sensor. The `Safety` node answers with an acknowledgement that includes as parameter the estimated value of the motor position. Both nodes compute the curve of the discrete positions that must be reached by the robot arm, depending on the initial value and the movement command. Any difference between the calculated values (or no data at all) implies an anomaly in the function of the robot movement (or communication link), which triggers the stop of the robot and the generation of an emergency signal (both nodes have access to the actuator).

Element from Fig. 3	Ada implementation	Note
Node	Task	Does its main function (control, monitor and MMI).
Watchdog	Task	Synchronous rendezvous with timeout.
Access to hardware	Protected object	Periodically updated by a task.
Real time issues	—	Both node and watchdog are periodic tasks. Watchdog has higher priority.

Table 4. Relation between deployment diagram and Ada objects

4.3 Safety Conclusions for the Crane Robot

Although only the study of safety for the primary positioning system has been presented, in this last subsection we want to present a summary of the conclusion that can be extracted of the whole study. To do so, the 31 identified hazards have been classified in six groups, depending on the type of safeguard adopted. The following conclusions can be extracted from this study:

- 45% of the safety requirements do not affect neither the design of the architecture nor its possible evolution.
- 55% of the safety requirements do affect architecture:
  - 40% imply the addition or extension of some components so that the values of the actuators can be double–checked.
  - 6.66% imply the design and adoption of redundant nodes as the one described in subsection 4.2.
  - 8.66% imply the addition of new sensors to the robot to monitor the system (generally, safety–related sensors).

These extensions or additions to the basic architecture (based only on the functional requirements) due to the safety requirements, mean the need of making cross verifications in practically every level of the architecture, which makes the process of designing the architecture harder and more complicated.

5 Conclusions and Future Works

It is always desirable to make the analysis of the possible hazards for any system to improve its design and safety. When the system interacts with the environment and/or with humans (as project EFTCoR does), the analysis becomes indispensable. But the analysis of hazards is a complex process that needs the support of a methodology. The more domain–specific the methodology, the more accurate the results will be. We have used the ANSI/RIA standard as the basis for the identification and classification of the hazards, risks and the safeguards to be adopted to reduce the risks to acceptable levels. This standard can be complemented by the safety patterns extracted from Douglass when designing a more complex solution and the use of fault trees to identify the possible causes of failure. In this sense, we hope that soon an European standard, derived from EN 61508 and specifically targeted to robotics systems, soon appears to fulfil the lack of a methodology for safety requirements specification and solutions in the EU.

Although it may seem that this work is the result of applying together ("glued" even) several standars, the contribution of this work goes further on because:

1. It gathers the methodologic experience of diverse authors, since this experience is usually absent in most of the standards.

2. The range of application of the proposal is wider than that of one of a single standard or technique seen in subsection 3.1, because this work covers from requirements specification to the implementation patterns applied in architectural design.
3. Lastly, a case study of a real application has been presented, where the safety requirements were naturally present from the beginning of the proyect, not added later.

From the work on safety requirements for the crane robot two important conclusions can be extracted: (1) only half of the safety requirements really affect the software architecture of the system and (2) only a few fraction of them require the use of external redundant control that must conform to the strictest level of safety. Nevertheless, since security requirements are, conceptually, independent of the functional ones, it would be more than desirable to have an architectural approach that allows this conceptual separation of concerns could be used by the designer. This is the line of work we are currently working on in the context of the research project DYNAMICA with the Universidad Politécnica de Valencia (Spain) and its ADL, PRISMA. It's also necessary to have a proper methodology to extract the safety requirements from the tables of risks and hazards and to have tools to catalogue them and to track their use and ease their reuse in another products of the same family, which is the aim of that project also shared with the University of Murcia in Spain.

References

1. P. Neumann. *Computer-Related Risks.* Addison-Wesley Professional, October 1994. ISBN: 0-201-55805-X.
2. C. Fernández, A. Iborra, B. Álvarez, J.A. Pastor, P. Sánchez, J.M. Fernández, and N. Ortega. Co-operative Robots for Hull Blasting in European Shiprepair Industry. November 2004. ISSN: 1070-9932.
3. EFTCoR Official Site. `http://www.eftcor.com/`.
4. ANSI/RIA R15.06: american national standard for industrial robots and robot systems safety requirements. Robotic Industries Association, 1999.
5. Bruce Powel Douglass. *Doing hard time: developing real-time systems with UML, objects, frameworks and patterns.* Object Technology. Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN: 0-201-49837-5.
6. Nancy Leveson. *Safeware: system safety and computers.* ACM Press, New York, NY, USA, 1995. ISBN: 0-201-11972-2.
7. EN 61508: functional safety of electrical/electronic/programmable electronic safety-related systems. European Committee for Electrotechnical Standardization, 2003.
8. K. Hansen, A. Ravn, and V. Stavridou. From safety analysis to software requirements. 24(7):573–584, July 1998.